Sujet d'informatique

Autour de données météorologiques

Partie I

Variations autour du minimum

Dans toutes cette partie, on s'interdit l'usage de la fonction `min` préprogrammée en Python et permettant d'obtenir directement le minimum d'une liste donnée en argument. En revanche, on se donne la fonction `mini` suivante, écrite en Python.

```
1  def mini(t):
2      '''Calcule le minimum d'un tableau d'entiers ou de flottants.'''
3      if len(t) == 0:
4          return None
5      p = t[0]
6      for i in range(len(t)):
7          if t[i] <= p:
8              p = t[i]
9      return p
```

Q1. Expliquer le déroulement pas à pas (évolution de la valeur des variables) lors de l'appel `mini([6,2,15,2,15])`, puis donner la valeur renvoyée.

Q2. Prouver que lorsque `t` est une liste non vide d'entiers ou de flottants, `mini(t)` renvoie la valeur minimale des éléments de `t`. On pourra exhiber un invariant de boucle précis.

Q3. Évaluer la complexité temporelle de l'appel `mini(t)` en fonction du nombre n d'éléments de `t`

Q4. Proposer une modification de la fonction `mini` pour que la valeur renvoyée soit le *maximum* et non le *minimum*. On pourra utiliser la numérotation des lignes pour préciser le lieu d'éventuelles modifications et ainsi éviter de réécrire toute la fonction.

On souhaite récupérer non plus le minimum d'une liste mais la (une) position dans le tableau où le minimum est atteint. Dans l'exemple vu plus haut, il y a deux positions où ce minimum est atteint : `1` et `3`.

Q5. Expliquer le principe d'une fonction réalisant cette opération et en particulier le rôle des variables manipulées.

Q6. *Cette question ne sera lue que si la question précédente a été traitée.*
Écrire une fonction `position_mini` réalisant effectivement cette opération.

Q7. Préciser l'indice renvoyé si le minimum est présent plusieurs fois dans la liste.
Proposer une modification permettant de changer ce comportement.

On souhaite maintenant déterminer la valeur minimale d'un tableau bidimensionnel d'entiers/de flottants. Un appel de cette fonction pourrait être :

```
>>> mini2D([[10,3,15],[5,13,10]])
3
```

Q8. Expliquer le principe d'une fonction réalisant ce travail.

Q9. Programmer effectivement cette fonction (on supposera les listes internes de taille non nulle).

Q10. Évaluer la complexité temporelle de cette fonction.

On souhaite, partant d'une liste constituée de couples[1] (chaîne, entier), déterminer la/une chaîne pour laquelle l'entier/le flottant associé est minimal :

```
>>> chaine_mini([['Tokyo',7000],['Paris',6000],['Londres',8000]])
'Paris'
```

Q11. Écrire une fonction `chaine_mini` réalisant effectivement cette opération.

Q12. Écrire enfin une fonction `majores_par` prenant en entrée une liste `t` d'entiers/de flottants ainsi qu'un entier/flottant `seuil` et renvoyant le nombre d'éléments de `t` majorés (au sens strict) par `seuil` :

```
>>> majores_par([12,-5,10,9],10)
2
```

Partie II

Recherche dans les archives de Météo-France

Vous avez décroché un stage à Météo-France pour votre TIPE et on vous offre l'accès à leur base de données concernant l'année écoulée. Celle-ci est constituée de deux tables :

– la table `villes` stocke les informations sur chaque ville de France et contient les colonnes (attributs) suivantes:

`insee`: Chaîne de caractères représentant le numéro INSEE (caractéristique : deux villes différentes ont toujours deux numéros INSEE différents).

`nom`: Chaîne de caractères représentant le nom de la ville.

`dpt`: Chaîne de caractères[2] représentant le département de la ville.

`lat`: Flottant représentant la latitude de la ville en degrés (positif si la ville est dans l'hémisphère nord). On prendra 47° de latitude Nord comme étant le « milieu » de la France.

`lon`: Flottant représentant la longitude de la ville en degrés (positif si la ville est à l'Est du méridien de Greenwich). On prendra 2° de longitude Est comme étant le « milieu » de la France.

`pop`: Entier représentant la population de la ville.

[1] Ou liste de listes à deux éléments.

[2] Et non un simple entier du fait des départements corses « numérotés » 2A et 2B.

Extrait de la table `villes`

insee	nom	dpt	lat	lon	pop
2B 033	Bastia	2B	42.7	9.45	42 912
23 067	Saint Denis	23	45.7	2.25	766
25 056	Besançon	25	47.23	6.02	115 879
46 175	Saint Denis	46	44.62	1.98	940
67 482	Strasbourg	67	48.58	7.75	761 042
93 066	Saint Denis	93	48.93	2.35	107 762
97 411	Saint Denis	97	-20.87	55.43	145 347
...					

– la table `mesures` enregistre l'évolution des températures au cours de l'année et contient les colonnes (attributs) suivantes :

`ville`: Chaîne de caractères représentant le numéro INSEE identifiant la ville.

`jour`: Entier (de valeur comprise entre 1 et 365 pour l'année 2013) représentant le jour j de l'année où est faite la mesure (1 correspond au premier janvier alors que 365 correspond au 31 décembre).

`Tmin`: Flottant représentant la température minimale mesurée le jour j en degrés celsius.

`Tmax`: Flottant représentant la température maximale mesurée le jour j en degrés celsius.

Extrait de la table `mesures`

ville	jour	Tmin	Tmax
25 056	110	5.2	9.7
97 411	110	25.4	36.8
25 056	216	16.7	33.9
67 482	363	-2.7	3.3
...			

Q13. Définir ce qu'est une clef primaire. Expliquer si la colonne (attribut) `nom` de la première table peut être une clef primaire et, dans le cas contraire, si une autre colonne (attribut) peut jouer ce rôle. Même question pour la seconde table concernant la colonne (attribut) `ville`.

Q14. Écrire une requête en langage SQL qui récupère depuis la table `mesures` les relevés dont la température moyenne $\mathrm{T_{moy}} = \dfrac{\mathrm{T_{min}} + \mathrm{T_{max}}}{2}$ est strictement inférieure à 0°C (avec toutes les colonnes disponibles). On précise que dans le langage de l'algèbre relationnelle, la requête s'écrit

$$\mathrm{R}_1 = \sigma_{\frac{\mathrm{T_{min}}+\mathrm{T_{max}}}{2}<0}(\texttt{mesures})$$

Q15. Écrire une requête en langage SQL qui récupère depuis la table `villes` le numéro INSEE, le nom et le département de toutes les villes du quart nord-est de la France. On précise que dans le langage de l'algèbre relationnelle, la requête s'écrit

$$\mathrm{R}_2 = \pi_{\texttt{insee,nom,dpt}}(\sigma_{\texttt{lat}>47}(\texttt{villes}) \cap \sigma_{\texttt{lon}>2}(\texttt{villes}))$$

Q16. On considère la requête suivante[3] écrite dans le langage de l'algèbre relationnelle

$$\mathrm{R}_1 \bowtie_{\texttt{ville=insee}} \mathrm{R}_2$$

Préciser le nom et la signification de l'opération de l'algèbre relationnelle utilisée. Conclure sur ce que va renvoyer la requête précédente. Traduire la requête en langage SQL.

[3] On utilise aussi parfois la notation $\mathrm{R}_1[\texttt{ville} = \texttt{insee}]\mathrm{R}_2$

Partie III

Un brin d'analyse numérique

1 Manipulation des données

Vous avez à présent fini votre séjour à Météo-France mais vous avez pu sauvegarder précieusement dans un fichier nommé `besancon_2013.txt` les relevés météos concernant la ville de Besançon pour l'année 2013. Ce fichier contient 365 lignes (une pour chaque mesure et donc jour de l'année) du type[4]

Extrait du fichier `besancon_2013.txt`

```
# Jour;Tmin;Tmax
1;2.1;7.6
2;2.3;4.9
3;-1.9;5.7
...
168;16.7;32.3
169;18.8;32.0
...
365;-3.2;1.9
```

Sur chaque ligne, la chaîne de caractère correspond à trois champs séparés par des point-virgules, à savoir
- le numéro correspondant au jour de la mesure (entier naturel);
- la température minimale mesurée ce jour (en degrés celsius, flottant);
- la température maximale mesurée ce jour (en degrés celsius, flottant).

Pour lire des fichiers de ce type, vous écrivez la procédure suivante :

Fonction de lecture du fichier de données et appel effectif

```
1  def lecture_fichier(fichier):
2      f = open(fichier,mode='r')
3      jours,Tmin,Tmax = [ ],[ ],[ ]
4      for ligne in f:
5          if ligne[0] != '#':
6              t,T1,T2 = ligne.split(';')
7              jours.append(int(t))
8              Tmin.append(float(T1))
9              Tmax.append(float(T2))
10     f.close()
11     return jours,Tmin,Tmax
12
13 jours,Tmin,Tmax = lecture_fichier('besancon_2013.txt')
```

On rappelle que `append` rajoute l'élément donné en argument à la fin de la liste sur laquelle on l'applique et voici ci-dessous le descriptif de l'aide Python concernant l'action de `split` sur une chaîne de caractères.

Aide Python concernant `split`, méthode agissant sur une chaîne de caractères

```
|  split(...)
|      S.split(sep=None, maxsplit=-1) -> list of strings
```

[4] NB : les trois petits points ne sont bien sûr pas présents dans le fichier mais signalent juste que l'on a omis de recopier un certain nombre de lignes.

```
    |
    |      Return a list of the words in S, using sep as the
    |      delimiter string.  If maxsplit is given, at most maxsplit
    |      splits are done. If sep is not specified or is None, any
    |      whitespace string is a separator and empty strings are
    |      removed from the result.
```

Q17. Proposer un ordre de grandeur du nombre d'octets utiles du fichier `besancon.txt`.

Q18. Décrire à quoi servent les lignes 5 et 6 du programme précédent. Donner le type des variables `jours`, `Tmin` et `Tmax`. Donner ensuite, si cela a un sens, le type des objets contenus dans `jours`, `Tmin` et `Tmax`.

Q19. Expliquer pourquoi il est préférable d'utiliser `jours.append(int(t))` (ligne 7) plutôt qu'une concaténation du type `jours = jours + [int(t)]`.

Q20. Écrire une fonction `moyenne` qui prend en entrée deux listes `a` et `b` de mêmes tailles (condition que la fonction devra vérifier préalablement) et renvoie une liste de même taille contenant dans la case d'indice `i` la valeur moyenne des valeurs des flottants stockés dans les deux listes `a` et `b` à l'indice `i`.

Q21. En appliquant la fonction précédente, écrire l'instruction Python qui stocke dans la variable `Tmoy` la liste des températures moyennes journalières à partir des données stockées dans les listes `Tmin` et `Tmax`.

Q22. On considère qu'il est nécessaire de couper les arrivées d'eau extérieures pour risque de gel quand la température moyenne sur la journée est strictement inférieure à 0°C. En utilisant une des fonctions programmées dans la première partie, stocker dans la variable `nb_jours_gel` le nombre de jours où il a fallu couper l'eau des conduites extérieures pour la ville de Besançon.

2 Modélisation physique

Votre binôme de TIPE veut essayer de modéliser les variations annuelles de température en les couplant aux variations saisonnières d'ensoleillement. Vous n'avez pas parfaitement compris ses arguments à base de « corps noir » et de « loi de Stéfan », mais il semble assez sûr que l'équation[5] régissant l'évolution de la température (notée θ car exprimée en degrés Celsius) s'écrive

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} + \alpha\,[\theta(t) + \mathrm{T}_0]^4 = \mu\,[1 + \varepsilon\,\cos(\omega t)]$$

Vous écrivez donc la fonction suivante qui permet de résoudre numériquement des équations différentielles du type $y' = f(y,t)$ en utilisant la méthode d'Euler[6].

Implémentation de la méthode d'Euler en Python

```
1  def euler(f,y0,dt):
2      y,t,tmax = y0,0,950
3      y_arr,t_arr=[y],[t]
4      while t < tmax:
5          y = y + f(y,t)*dt
6          t = t + dt
7          y_arr.append(y)
8          t_arr.append(t)
9      return t_arr,y_arr
```

[5]Pour le lecteur curieux, on signale que dans cette équation : T_0 permet de passer des celsius aux kelvins, α s'apparente (à une constante près) à la constante σ de la loi de Stéfan de rayonnement du corps noir en $\sigma\,\mathrm{T}^4$, μ s'apparente (à une autre constante près) à la puissance moyenne du rayonnement solaire au niveau de la Terre et ε correspond au pourcentage annuel de variation de ce rayonnement de sorte que ω vaille $2\pi/365{,}25$ en rad/jour, ce qui donne bien une période d'un an.

[6]La fonction `append` a déjà été utilisée/discutée précédemment.

```
10
11 def f(theta,t):
12     return                    # à completer...
```

Q23. Expliquer le principe de la méthode d'Euler. On signalera notamment à quoi correspondent les différentes parties du code fourni pour la fonction `euler` et la nature et la signification des paramètres d'entrée `f`, `y0` et `dt`

Q24. En supposant que votre binôme ait stocké dans les variables *globales* `alpha`, `T0`, `mu`, `epsilon` et `omega` les valeurs adéquates pour le problème posé, compléter la fonction `f` pour que l'appel à la fonction `euler` renvoie effectivement une solution numérique de l'équation différentielle régissant l'évolution de θ.

Q25. Le graphe suivant représente les points expérimentaux (reliés par des traits pleins) auxquels on a superposé trois courbes intégrées avec une valeur différente du « pas d'intégration », respectivement de 27 jours, 54 jours et 81 jours. Associer à chaque pas d'intégration son symbole (rond noir, triangle bleu ou losange rouge) en justifiant. On expliquera en plus le comportement observé des symboles ronds lorsque le temps d'intégration est important.

Avertissement : La question bonus suivante ne fait appel qu'à un peu de bon sens et non au cours de physique ou de SI de CPGE.

Q26. Expliquer l'allure de la courbe expérimentale comparée à la modélisation. Si l'écart au modèle est caractérisé par une variable aléatoire, indiquer quelle valeur son espérance semble avoir d'après le relevé. Proposer l'allure de cette courbe expérimentale sur plusieurs années si on prend pour hypothèse l'existence d'un réchauffement climatique. Indiquer si ce dernier semble apparaître dans le relevé fourni et/ou dans la modélisation.